# 環境の変動に自律追従する省電力情報収集基盤

西谷 信之介[†1] 廣津 登志夫[†1] 福田 健介[†2]
菅原 俊治[†3] 栗原 聡[†4]

[†1] 豊橋技術科学大学 [†2] 国立情報学研究所 [†3] 早稲田大学 [†4] 大阪大学

生活環境の情報支援においては，配置の自由度などの観点で無線によるセンサネットワークの利用が期待されている．しかし，センサネットワークを構成するセンサノードの多くはバッテリーで動作することから，限られたバッテリーで効率よく観測環境の情報を収集するためには省電力情報収集基盤が必要である．本研究では観測環境の変動に着目し，観測対象の変動に追従してサンプリング間隔を制御することで，情報の質を落とさずにセンサネットワークの長寿命化を図る情報収集基盤を提案する．本稿では，情報収集基盤の基となる変動に自律追従するアルゴリズムの提案を行いその評価を示す．

# Power Saving Information Gathering System with Autonomic Fluctuation Tracking

Shinnosuke NISHITANI[†1] Toshio HIROTSU[†1]
Kensuke FUKUDA[†2] Toshiharu SUGAWARA[†3]
and Satoshi KURIHARA[†4]

[†1]Toyohashi University of Technology [†2] National Institute of Information
[†3] Waseda University [†4] Osaka University

On an information support of an living environment, we can expect to use wireless sensor networks because it can be freely deployed. However, a number of sensor nodes which build the wireless sensor networks are required to spend the energy cost. An energy-efficient data gathering system is needed to collect the environmental information effectively with the limited power source(such as batteries) at the sensor node. We focus on fluctuations of the environment, and propose the power saving information gathering system which enhances life-time and maintains the data quality. Our approach is to control data sampling at the instant corresponding to the fluctuation of the environment. In this pepar, we perform an evaluation of the autonomic algorithm for the power saving information gathering systemwith autonomic fluctuation tracking.

## 1 はじめに

近年，無線通信技術の向上とともにその技術を用いた無線通信機器の小型化が進み，センシング，無線通信，データ処理・演算機能を持ち合わせたセンサノードと呼ばれる小型機器が開発されている．そしてセンサノードを多数配置してネットワークを構成することにより生活環境を監視，観測，制御する無線センサネットワーク技術が注目を集めている．

センサノードは多くの場合，バッテリーで動作するため，センサネットワークの連続運用にはある程度の時間的な上限が存在する．そのため環境内に配置された多数のセンサノードのバッテリーを頻繁に交換することは現実的に困難であることから，センサネットワークの長期間運用を可能にする電力効率の良いセンサ情報収集基盤は重要である．

センサネットワークの長期間の運用を可能にするには，個々のセンサノードの省電力化が重要である．そこで本研究では，環境の変動に応じてセンサネットワークを構成する個々のセンサノードが自律的にサンプリング間隔を制御することで，センサネットワークの長寿命化を図り，一定のサンプリング間隔で得られる情報に比べて質を落とさない省電力情報収集基盤を提案する．

以降，2章で自律的に環境の変動に追従する情報収集基盤を提案する．3章で観測環境の変動に追従するアルゴリズムを示し，4章でそのアルゴリズムの評価を行う．

## 2 環境変動に追従する省電力情報収集基盤

センサノードの電力消費を抑えるために考慮すべきことは，次のことが挙げられる．

1. センサノード間の無線通信距離の短縮

2. 送受信する情報量の削減
3. センサデータ送受信頻度の削減

これらのことに対して，取り組んでいる研究は多数ある．送受信する情報量の削減とセンサノード間の無線通信距離の短縮に取り組んでいる研究としては，LEACH[1] がある．この研究はセンサノードをクラスタリングすることで，センサデータの集約と無線通信距離を短縮し，省電力化を図っている．また，LEACH と同様にクラスタを作成し，省電力化を図る手法として，CAG[2] が挙げられる．LEACH との違いはセンサデータの相関性を利用してクラスタを作成し，その相関性を利用してセンサデータを集約することで送受信する情報量を削減する．

一方，サンプリング間隔を制御してセンサデータの送受信頻度を削減する研究として，[3],[4] がある．[3] は，観測対象のモデルを用意し，そのモデルと観測値との誤差をフィードバック制御系に与え，サンプリング間隔を制御する手法を提案している．[4] は，カルマンフィルターを利用して，与えられたサンプリング間隔内で自律的にサンプリング間隔を制御する．サンプリング間隔の範囲は，センサノードからの観測値を受け取るサーバによって決定され，センサノードのサンプリング間隔がその範囲を超えると，新たにサンプリング間隔の範囲がサーバより与えられる．これらの研究では，サンプリングデータの精度を維持しながらサンプリング頻度や送受信頻度を制御するために，集中制御型のサーバが存在している．しかし，多数のセンサノードが連携して稼働するセンサネットワーク環境においては，環境に追従するように各センサノードがある程度自律的に制御できることが望ましい．そこで，本研究ではセンサノードが収集したデータの変動に対して自律的にサンプリング頻度を調整し，電力消費を抑制しつつ取得データの質を維持する手法について検討し，実際のセンサノードから取得した情報を用いて評価を行った．

## 3 観測値に追従するアルゴリズム

数値的変動に対してサンプリング間隔を制御し，追従するアルゴリズムを提案する．追従アルゴリズムは，パラメータを変更することで，特性の異なる観測環境のセンサ変動特性に対応する．

### 3.1 追従アルゴリズム

センサノードが観測値を観測すると，追従アルゴリズムにより，次にセンサノードが動作するまでの時間，つまりサンプリング間隔を計算する．

図 1 に追従アルゴリズムを示す．まず観測された観測値 value をバッファ長 $N$ のバッファにバッファリングする．そのバッファ内の観測値の平均 bufAvg を基準に，その標準偏差 bufSD のプラスマイナスを閾値とし，観測値がその閾値以内なら変動なし，それより以外なら変動ありとして，観測値が変動しているかを判定する（1 行目）．バッファ内の観測値の平均 bufAvg と標準偏差 bufSD は式 1 と式 2 より求める．

$$\mathrm{bufAvg} = \frac{\sum_{i=1}^{N} \mathrm{value}_i}{N} \tag{1}$$

$$\mathrm{bufSD} = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{value}_i - \mathrm{bufAvg})^2}{N}} \tag{2}$$

1 行目の thresh 関数は，変動の判定に利用する閾値を返す関数で，引数として bufSD を取る．

1 行目で変動ありと判定されれば，サンプリング間隔を狭め（2〜7 行目），変動なしと判定されれば，サンプリング間隔を広げる（9〜14 行目）．それぞれ fineSamples 関数でサンプリング間隔を狭め，coarseSamples 関数でサンプリング関数を広げる．それぞれの関数も引数として bufSD をとる．thresh 関数，fineSamples 関数，coarseSamples 関数は，観測環境のセンサ変動特性にパラメータを設定する．

また，サンプリング間隔を保証するために，サンプリング間隔の最小値 INTERVAL_MIN と最大値 INTERVAL_MAX を設けて，それ以下やそれ以上にならないようにする．この値も観測環境のセンサごとの変動特性によって変更する．

以上で挙げた 3 つの関数はそれぞれ引数として bufSD を取る．bufSD は観測値が観測されるたびに計算されるため，観測値の変動に応じてそれぞれの値も変動する．bufSD の値が大きいときは，観測値が大きく変動していることが分かる．そのため，変動に応じてそれぞれの関数の返値を変えることができる．例えば，閾値を設定する thresh 関数において，bufSD が小さいときは閾値を大きく設定することで小さな変動では fineSamples 関数を呼ばないする．逆に bufSD が大きいときは閾値を小さく設定して，小さな変動でも fineSamples 関数を呼ぶようにし，変動が大きい観測値をより密に観測可能になる．同様に fineSamples 関数や coarseSamples 関数においても，bufSD が大きいときはサンプリング間隔を大きく狭めたり，広げたりというように，変動に応じてサンプリング間隔を制御することが可能である．

```
1: if ((bufAvg + thresh(bufSD)) ≥ value && (bufAvg
   - thresh(bufSD)) < value) then
2:   incTime = fineSamples(bufSD)
3:   if ((samplingTime + incTime) ≥ INTER-
     VAL_MAX) then
4:     samplingTime = INTERVAL_MAX
5:   else
6:     samplingTime += incTime
7:   end if
8: else
9:   decTime = coarseSamples(bufSD)
10:  if ((samplingTime - decTime) ≤ INTER-
     VAL_MIN) then
11:    samplingTime = INTERVAL_MIN
12:  else
13:    samplingTime -= decTime
14:  end if
15: end if
```

図 1: 追従アルゴリズム

### 3.2 各センサへの対応

ここでは観測対象の観測値の変動が時間的に速いセンサと遅いセンサに対応する thresh 関数，fineSamples 関数，coarseSamples 関数を定義する．それぞれの関数は bufSD の一次関数とする．それらの一次関数の傾きを coefficient，切片を intercept とする．

#### 3.2.1 変動の遅いセンサ

- thresh(bufSD)
  変動が遅いので，標準偏差以上に変動していれば，変動と判別する関数とした．
  coefficient * bufSD
- fineSamples(bufSD)
  標準偏差が大きければ大きいほど，サンプリング間隔を狭める関数を用いた．以下のような関数である．
  SAMPLING_MIN * coefficient * bufSD
- coarseSamples(bufSD)
  標準偏差が小さいときはサンプリング間隔を多めに広げ，大きいときは小さめに広げる関数を用いた．以下のような関数で，負の場合は 1 を返す．
  SAMPLING_MIN * (−coefficient * bufSD + intercept)

#### 3.2.2 変動の速いセンサ

- thresh(bufSD)
  標準偏差が大きいときは変動があることを示しており，その場合は少しの観測値の変化でも変動であると判定すると良いと考えられる．そのため，bufSD が大きければ，細かい変動でもサンプリング間隔を狭め，密にセンシングするような以下の関数とした．
  (−coefficient * bufSD + intercept)
- fineSamples(bufSD)
  変動の速いセンサの場合は変動があった直後に，再び変動がある場合が考えられる．それに対応するために，急激にサンプリング間隔が狭まる関数とした．
  SAMPLING_MIN * coefficient * bufSD
- coarseSamples(bufSD)
  サンプリング間隔が広がりにくい関数とするために，coefficient は 1 以下とする．
  coefficient * SAMPLING_MIN

## 4 評価

まず，予備評価としてサンプリング間隔の変化による電力消費を測定する．次に，サンプリング間隔による測定値の質の違いを調べた上で，前節で提案した変動に追従するアルゴリズムの評価を行う．

### 4.1 電力消費に関する予備実験

本提案手法の効果を知るためには，サンプリング間隔の制御がどの程度，電力消費に関わってくるかを知る必要がある．そこで，指定した間隔でセンサデータを取得しシンクノードに送信するアプリケーションを MICAz[5] 上に実装し，サンプリング間隔の変化に対する電力消費の違いについて調べた．

#### 4.1.1 実験内容

センサノードには MICAz を使用し，その上に載せるセンサボードは MTS310CB[5] を用いた．MTS310CB は光度センサ，音圧センサ，温度センサ，2 軸加速度センサ，2 軸磁力センサなどの各種センサが実装されている．実験は MICAz 上の光度センサとバッテリー電圧の値を取得後，シンクノードに送信するアプリケーションを実装し，そのサンプリング間隔を 0.1 秒，1.0 秒，10 秒，60 秒と変化させ，バッテリーの電圧降下で電力消費の違いについて調べた．そのアプリケーションをそれぞれのサンプリング間隔で 3 日間動作させ，電圧降下の違いについて調べ

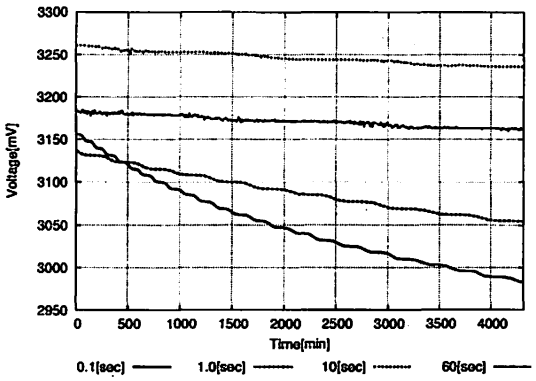



図 2: バッテリー電圧

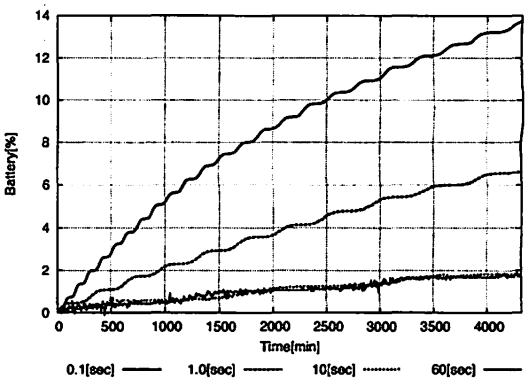



図 3: バッテリー消費率

た．なおバッテリーは新品の単 3 アルカリ乾電池を用いた．

#### 4.1.2 実験結果

図 2 と図 3 に実験結果を示す．図 2 は各サンプリング間隔ごとのセンサノードのバッテリー電圧を示し，図 3 はその測定値から求めたバッテリーの消費率である．消費率は式 3 により求め，$V_0$ はバッテリーの初期電圧，$V_i$ は現在の電圧，$V_{low}$ は MICAz の最低動作電圧を示す．式 3 は MICAz の最低動作電圧に対して，現在のバッテリー電圧が何パーセント減少したかを示している．したがって，消費率が 100%に到達するとセンサノードが停止することを意味する．ここで用いた MICAz の最低動作電圧 $V_{low}$ は 1892[mV] で，これは 5 台の MICAz が通信不能になったときの電圧値の平均値である．

図 2 より，サンプリング間隔の違いにより，電力消費が大きく異なることが分かる．これより環境の変動に応じて，センサノードのサンプリング間隔を制御することは非常に有効な方法であることが分かる．また，図 2 より，同じ銘柄の新品の電池と同じ機器を使っているのにもかかわらず，初期電圧に大きな違いが出ている．同じ電池でもセンサノードの電圧の測定値に差があったため，これはセンサノードの個体差によるものである．

$$\left(1 - \frac{V_i - V_{low}}{V_0 - V_{low}}\right) \times 100 \qquad (3)$$

### 4.2 サンプリング間隔の違いに関する調査

本研究で提案した変動に追従するアルゴリズムを評価するために，まず性質の異なるセンサにおいて，サンプリング間隔を固定した場合に，どのようなセンサデータが取得されるかを調査する必要がある．

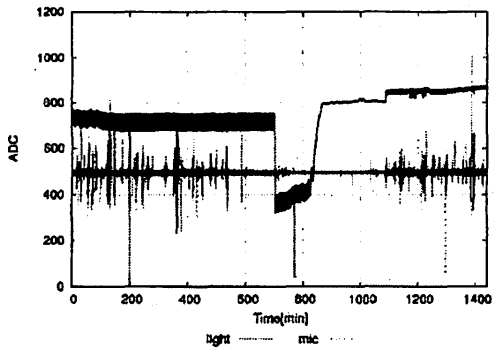
センサデータの取得には MICAz と MTS310CB を用い，変動の遅いセンサを光度センサ，変動の速いセンサを音圧センサと想定して，部屋の 24 時間観測を行った．調査はサンプリング間隔 0.1 秒で観測した後，そのデータから 1.0 秒，10 秒，60 秒のサンプリング間隔でデータを間引いて，サンプリング間隔 0.1 秒に対して，どのように変化するか比較を行う．

図 4 にそれぞれのサンプリング間隔で得られたセンサデータを示す．図 4 の縦軸はセンサボードから得られる AD 変換の値である．変動の遅い光度センサはサンプリング間隔を変えても波形に大きな変化は見られない．それに対して，変動の速い音圧センサはサンプリング間隔を変えると，波形の形が全く異なる．サンプリング間隔 60 秒ではほとんど変動が得られていない．

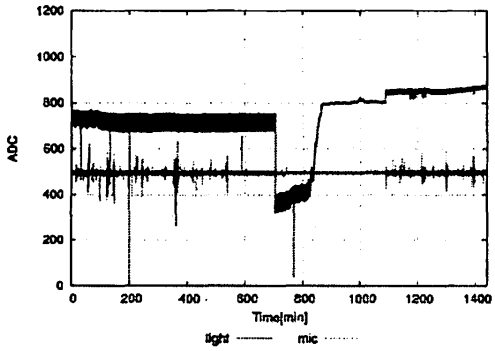
次に数値的な評価を行う．サンプリング間隔 0.1 秒のセンサデータとサンプリング間隔 1.0 秒，60 秒，600 秒のセンサデータとの違いを DP(Dynamic Programing) マッチングにより 2 乗距離を求め，比較を行った．距離は小さいほど良く，波形が似ていることを示す．その結果を表 3 に示す．光度センサと音圧センサではサンプリング間隔を広げたときのサンプリング間隔 0.1 秒に対する 2 乗距離の増え方が異なる．光度センサはサンプリング間隔を広げると，桁が変わるほど 2 乗距離が増加するが，音圧センサは桁が変わるほど増加しない．

### 4.3 追従アルゴリズムの評価実験とその評価

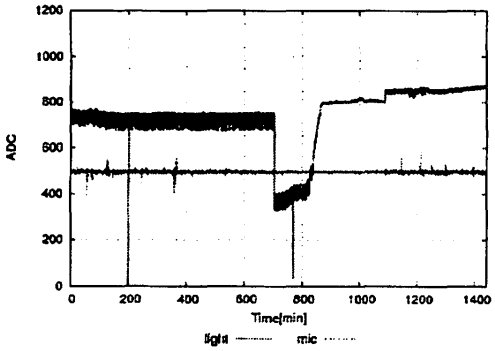
変動が遅いセンサには光度センサを選択し，また変動が速いセンサには音圧センサを選択し，追従アルゴリズムの評価を行う．各センサの特性に対応するためのパラメータは表 1 と表 2 のように設定した．各パラメータを設定したアルゴリズムで得られたセンサデータを図 5 に示す．変動の遅い光度センサの



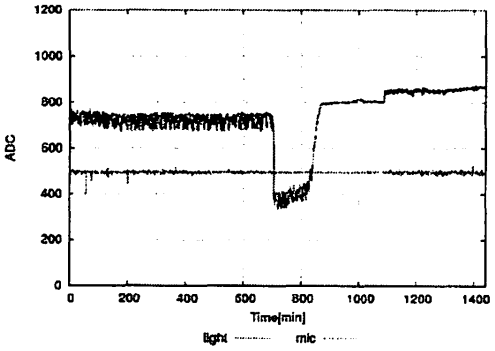

(a) サンプリング間隔 0.1 秒




(b) サンプリング間隔 1.0 秒




(c) サンプリング間隔 10 秒




(d) サンプリング間隔 60 秒

図 4: サンプリング間隔の違い

データはサンプリング間隔 0.1 秒の波形と比べても，波形は酷似している．また，変動の速い音圧センサ

表 1: 光度センサへの対応

| | coefficient | intercept |
|---|---|---|
| thresh() | 1.1 | – |
| fineSamples() | 30 | – |
| coraseSamples() | -1 | 50 |
| SAMPLING_MIN | 1.0 秒 | |
| SAMPLING_MAX | 600 秒 | |

表 2: 音圧センサへの対応

| | coefficient | intercept |
|---|---|---|
| thresh() | -2 | 20 |
| fineSamples() | 100 | – |
| coraseSamples() | 1 | – |
| SAMPLING_MIN | 0.1 秒 | |
| SAMPLING_MAX | 10 秒 | |

のデータはサンプリング間隔 0.1 秒のデータと比べて，いくつかの変動は取りこぼしているが，変動には追従できていると考えられる．

次に数値的な評価を行う．4.2 節の数値的な評価と同様に DP マッチングにより，サンプリング間隔 0.1 秒のセンサデータとの 2 乗距離を求め，評価を行う．また，24 時間分のデータに対する各サンプリング間隔と追従アルゴリズムのサンプリング回数の比較も行った．サンプリング回数が少ないほど，省電力であることを示す．それぞれ表 3 と表 4 に評価結果を示す．

表 4 より，追従アルゴリズムを用いたときの光度センサのサンプリング回数はおよそサンプリング間隔 60 秒のサンプリング回数より少ない．そのことから，電力消費では追従アルゴリズムの方が優れている．また，サンプリング間隔 60 秒の 2 乗距離と追従アルゴリズムの 2 乗距離を比べると，追従アルゴリズムの 2 乗距離の方が小さく，サンプリング間隔 60 秒より質が良いと考えられる．これらのことから，サンプリング間隔 60 秒に比べて，省電力でより質の高いデータが得られていると言える．次に音圧センサも同じように比較すると，音圧センサの追従アルゴリズムのサンプリング回数はおよそサンプリング間隔 1.0 秒のサンプリング回数に最も近い．そこから表 1 の 2 乗距離を比べると，追従アルゴリズムの方がサンプリング間隔 1.0 秒より 2 乗距離が大きく，あまり追従アルゴリズムの効果がなかったことが分かる．しかし，図 4(b) のサンプリング間隔 1.0 秒と図 5 の追従アルゴリズムの音圧データを比較すると，追従アルゴリズムの方がサンプリング回数が少ない

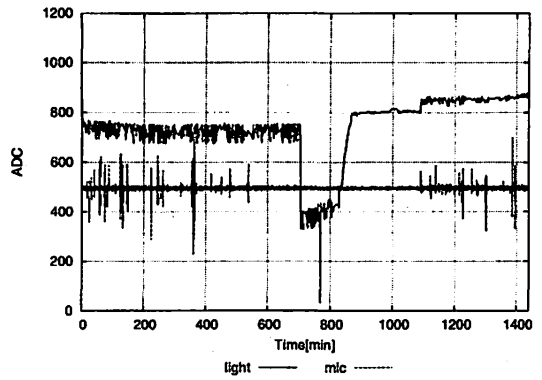



図 5: 追従アルゴリズム

表 3: サンプリング間隔 0.1 秒に対する 2 乗距離

| | 光度センサ | 音圧センサ |
|---|---|---|
| サンプリング間隔 1.0 秒 | 3.83E06 | 1.80E06 |
| サンプリング間隔 60 秒 | 9.25E06 | 2.15E06 |
| サンプリング間隔 600 秒 | 1.16E07 | 2.58E06 |
| 追従アルゴリズム | 9.16E06 | 1.99E06 |

表 4: サンプリング回数

| | 光度センサ | 音圧センサ |
|---|---|---|
| 追従アルゴリズム | 1272 | 24163 |
| サンプリング間隔 0.1 秒 | 864000 | |
| サンプリング間隔 1.0 秒 | 86400 | |
| サンプリング間隔 60 秒 | 1440 | |
| サンプリング間隔 600 秒 | 144 | |

にもかかわらず，同程度の変動が観測できている．

## 5 考察

変動の遅い光度センサに対して，追従アルゴリズムを適応した場合は，サンプリング間隔 1.0 秒よりも少ないサンプリング回数で，サンプリング間隔 0.1 秒に対する 2 乗距離が短くなり，より質の高いデータが得られた．このことはより少ない電力消費で，かつ質の高いデータが得られたと言うことである．例えば，光度センサに追従アルゴリズムを用いた場合のバッテリー消費を考えると，4.3 節より，追従アルゴリズムのサンプリング回数はサンプリング間隔 60 秒に最も近いので，図 3 のサンプリング間隔 60 秒の 1440 分 (1 日) 時点でのバッテリー消費率を見ると約 1%である．単純にバッテリー消費が一次関数であると仮定すると，約 100 日間動作することになる．同じ 100 日間動作し，より質の高いデータが得られるのならば，明らかに追従アルゴリズムを用いて観測を行った方がよいと考えられる．

しかし，変動の速い音圧センサに対して，追従アルゴリズムを適応した場合は，電力消費を抑えつつ，質の高い観測データを得られなかった．これに関しては，パラメータなどの調整が必要であると考えられる．

## 6 まとめ

本稿では，センサデータの変動に対して自律的に追従するアルゴリズムを提案した．その評価を行った結果，サンプリング間隔が一定の場合より，電力消費を抑えつつ，質の高いデータを得ることができた．

今後はパラメータの調整を進めた上で，自律追従アルゴリズムを MICAz 上に実装し，実環境での評価を行いたいと考えている．

### 参考文献

[1] W. R. Heinzalman, A. Chandrakasan, and H, Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proceedings of HICSS-33*,pp. 3005-3014, Jan. 2000.

[2] S.Yoon, and C.Shahai. "Exploitiing Spatial Correlation Towards and Energy Efficient Clustered AGgregation Technique (CAG)", In ICC, pp. 82-98, May 2005

[3] A. D. Marbini and L. E. Sacks. Adaptive sampling mechanisms in sensor networks. In London Communications Symposium, London, UK, 2003.

[4] A. Jain and E. Y. Chang. Adaptive sampling for sensor networks. Proc. DMSN' 04, pp. 10-16, 2004.

[5] http://www.xbow.com